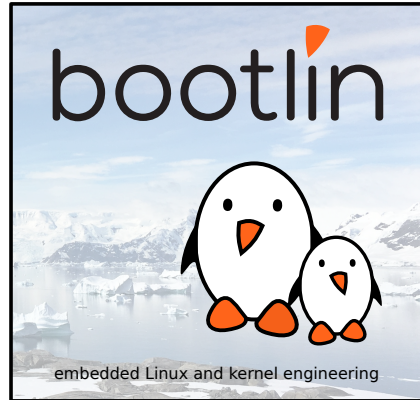




# RTC subsystem, recent changes and where it is heading

Alexandre Belloni  
*[alexandre.belloni@bootlin.com](mailto:alexandre.belloni@bootlin.com)*

© Copyright 2004-2019, Bootlin.  
Creative Commons BY-SA 3.0 license.  
Corrections, suggestions, contributions and translations are welcome!





- ▶ Embedded Linux engineer at Bootlin
  - ▶ Embedded Linux **expertise**
  - ▶ **Development**, consulting and training
  - ▶ Strong open-source focus
- ▶ Open-source contributor
  - ▶ Maintainer for the Linux kernel **RTC subsystem**
  - ▶ Co-Maintainer of **kernel support for Microchip ARM SoCs**

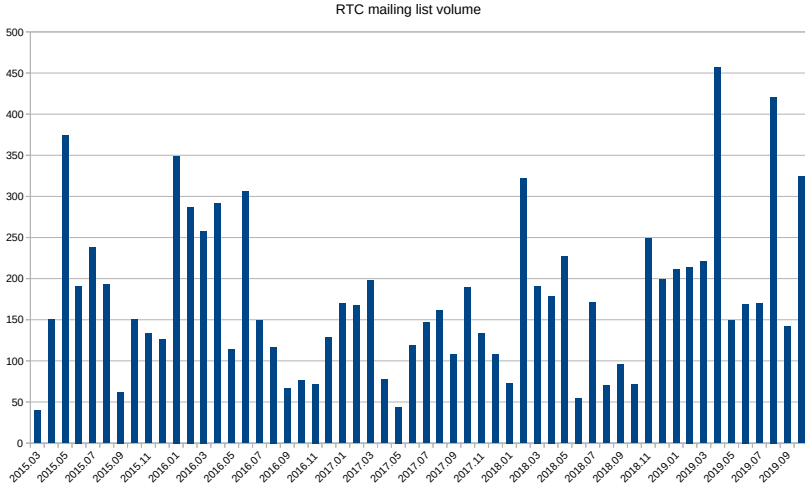


# The RTC subsystem

- ▶ Introduced in 2.6.17 (March 2006) by Alessandro Zummo
- ▶ Taken over by Alexandre Belloni since 4.1 (April 2015)
- ▶ Mailing list was on googlegroups until Google decided that it was sending to much spam on the 4th of May 2017.
- ▶ `linux-rtc@vger.kernel.org` created on 11th of May 2017
- ▶ Archives are at <https://lore.kernel.org/linux-rtc/>
- ▶ Patchwork at <http://patchwork.ozlabs.org/project/rtc-linux/list/>
- ▶ git repository on kernel.org:  
`git://git.kernel.org/pub/scm/linux/kernel/git/abelloni/linux.git`

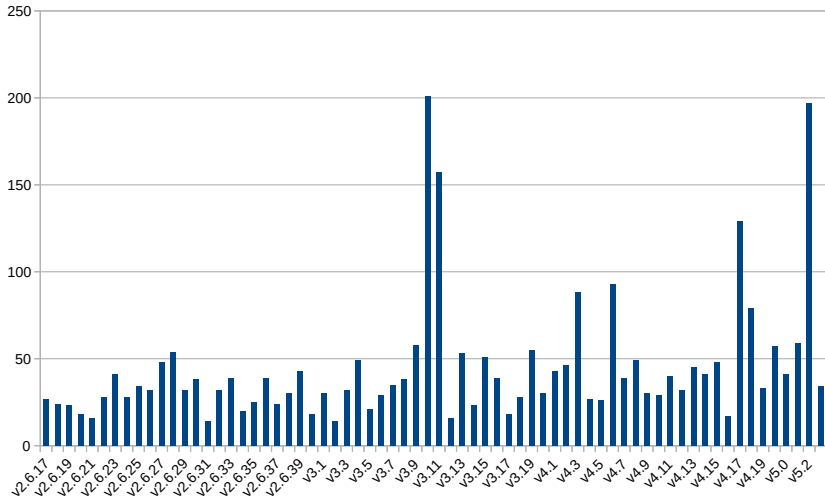


# Mailing list statistics





# Commits statistics





## Recent changes: new drivers since 4.0

- ▶ Abracon AB-RTCMC-32.768kHz-EOZ9
- ▶ Alphascale ASM9260
- ▶ Amlogic Meson RTC
- ▶ Amlogic Virtual Wake
- ▶ Android emulator (goldfish) RTC
- ▶ Aspeed BMC SoC RTC
- ▶ Broadcom STB wake-timer
- ▶ Cadence RTC IP
- ▶ Chrome OS EC RTC
- ▶ Cortina Gemini
- ▶ Dialog DA9062
- ▶ Epson RX6110SA
- ▶ Epson RX8010SJ
- ▶ Epson RX8130CE
- ▶ Epson Toyocom rtc-7301sf/dg
- ▶ Freescale FlexTimer Module alarm
- ▶ Freescale i.MX system controller RTC
- ▶ Intersil ISL12026
- ▶ Maxim IC DS1308
- ▶ Maxim MAX77620
- ▶ Maxim MAX6916
- ▶ Mediatek MT6397
- ▶ Mediatek MT7622 RTC
- ▶ Microchip PIC32
- ▶ Microcrystal RV1805
- ▶ Microcrystal RV3028
- ▶ Microcrystal RV8803
- ▶ Motorola CPCAP PMIC RTC
- ▶ NXP i.MX53 SRTC
- ▶ NXP LPC24xx
- ▶ NXP PCF2127/PCF2129
- ▶ NXP PCF85263/PCF85363
- ▶ Realtek RTD1295
- ▶ Spreadtrum SC27xx PMIC RTC
- ▶ STMicroelectronics STM32
- ▶ Whwave sd3078
- ▶ Xilinx Zynq MP



## Recent changes: crystal offset

- ▶ The RTC has either an in-package or external crystal, usually running 32kHz.
- ▶ There is variance in the exact frequency because of manufacturing, temperature and aging.
- ▶ Some RTCs can add or subtract correction pulses to make seconds quicker or longer.
- ▶ New sysfs interface `/sys/class/rtc/rtcX/offset`
- ▶ The unit is part per billion, positive value adds pulses and negative values subtracts pulses.
- ▶ The name is coming from the NXP datasheets, it is also called digital trimming.



## Recent changes: crystal offset

- ▶ An other way to accommodate crystal variance is to change the load capacitance.
- ▶ Some RTC oscillators can change the apparent load on X1/X2
- ▶ New device tree property `quartz-load-femtofarads`
- ▶ It is also called analog trimming.





## Recent changes: non volatile memory

- ▶ Some RTCs have a small amount of RAM.
- ▶ Because the RTC is always on, the RAM is non volatile.
- ▶ This memory is now exposed using the `nvmem` framework.
- ▶ New registration function: `int rtc_nvmem_register(struct rtc_device *rtc, struct nvmem_config *nvmem_config)`
- ▶ Multiple calls are allowed so if EEPROM is available, it can also be exposed through `nvmem`.
- ▶ `rtc->nvrwam_old_abi` handles the legacy ABI for drivers that were exposing a file named `nvrwam` in sysfs. It issues deprecation warning.



## Recent changes: rtc registration

- ▶ With `devm_rtc_device_register`, it was difficult to write a probe function without race conditions.
- ▶ The registration is now split in two parts: `devm_rtc_allocate_device` and `rtc_register_device`.
- ▶ This allows to handle the `struct rtc_device` before registering it.
- ▶ Non devm managed registration has been removed from the driver API.



## Recent changes: sysfs attributes registration

- ▶ sysfs attributes were created after `devm_rtc_device_register`. This opened up a possible race condition because the rtc folder appeared before the extra attributes.
- ▶ New functions: `int rtc_add_group(struct rtc_device *rtc, const struct attribute_group *grp)` and `int rtc_add_groups(struct rtc_device *rtc, const struct attribute_group **grps)`.
- ▶ Used between `devm_rtc_allocate_device` and `rtc_register_device`, they ensure all the RTC attributes are available at the same time.



## Recent changes: RTC range

- ▶ New members of `struct rtc_device`: `range_min` and `range_max`.
- ▶ They allow the driver to declare the date and time range supported by the RTC hardware.
- ▶ No discontinuity is allowed in this range.
- ▶ The core will check `struct rtc_tm` is in the range before passing it to the driver.



## Recent changes: RTC offset

- ▶ To solve RTC end of time issues, the core is now able to offset and window the hardware range.
- ▶ The offset is added or subtracted before calling the driver callbacks.
- ▶ Note that because the RTC time is always positive, there are almost no RTCs failing in 2038 (currently only 3). The most common failing date is 2100, followed by 2106.
- ▶ This is still useful to handle dates before 2000 as it seem to be mandatory for Android.
- ▶ New device tree property: `start-year`



## Recent changes: other improvements

- ▶ New `%ptR` printk format
- ▶ Tracepoints
- ▶ `time_t` overflow is now properly handled in `rtc_hctosys` to avoid breaking 32bit platforms.
- ▶ `rtc-range` tool to test the continuous range of an rtc git:  
`//git.kernel.org/pub/scm/linux/kernel/git/abelloni/rtc-tools.git`
- ▶ Timestamping
- ▶ Trickle charging



## Recent changes: cleanups

- ▶ Unnecessary `rtc_valid_tm` calls have been removed.
- ▶ `open`, `release`, `set_mmss` and `set_mmss64`, `read_callback` members of `struct rtc_class_ops` have been removed.
- ▶ `struct rtc_class_ops` are now `const` where possible.
- ▶ `rtc_control` API has been removed. It has been replaced by `hrtimer`.
- ▶ `rtc_irq_register` has been removed.
- ▶ Core files have been renamed for consistency.



## Future changes: voltage detection

- ▶ Current voltage drop detection ABI is not working well for more advanced RTCs.
- ▶ `RTC_VL_READ` returns a currently undocumented value.
- ▶ There are different types of monitored voltages: main supply, battery or auxiliary supply.
- ▶ They can have multiple states: OK, low, low and functionalities are disabled, data lost.
- ▶ A new ioctl has to be developed.





## Future changes: timestamping

- ▶ Some RTCs can store one or multiple timestamp when an event happens.
- ▶ Currently exposed through `/sys/class/rtc/rtcX/timestampX`
- ▶ This is open coded in each driver. There is room for factorization in the core.
- ▶ Timestamp events will probably need to be configured in the future (timestamp first event, last event, input pin).
- ▶ Not sure whether this will be using a new ioctl or more sysfs files.



## Future changes: backup switch mode

- ▶ It is possible to select the backup switch policy for some RTCs.
- ▶ The common policies are:
  - ▶ disabled
  - ▶ direct ( $VDD < VAUX$ )
  - ▶ standby
  - ▶ level ( $VDD < \text{threshold}$ )
- ▶ This will be implemented through a device tree property because it directly depends on the type of auxiliary voltage provided by the board.
- ▶ It is necessary to avoid hardcoding in the driver because it may have been set properly by the bootloader.



## Future changes: alarm handling

- ▶ Alarm support detection. Many drivers modify the `struct rtc_ops` when alarms are not present. This prevents constification.
- ▶ Alarms with minute granularity. Support is currently open coded in each driver.
- ▶ Wakeup support: many parameters have to be taken into account to know whether the RTC can wakeup the system, there is potential for factorization in the core.
- ▶ Alarm routing:
  - ▶ Some RTCs have multiple interrupt pins and can configure which interrupt goes to which pin.
  - ▶ This is useful to route alarms to a CPU (wakeup) or a PMIC (powerup)
  - ▶ Device tree properties will be used.



## Future changes: BCD RTCs

- ▶ Many RTCs store the date and time in BCD in a somewhat common format.
- ▶ There is potential for code factorization, especially when using regmap to access the RTC registers.



## Future changes: timers

- ▶ Some timers are now able to wakeup the platform.
- ▶ They don't actually handle the system time, either because the counter is too small or it is counting downward or it is impossible to read it.
- ▶ The drivers currently open code the dummy `read_time` and `set_time` to keep the RTC core happy.



## Future changes: other topics

- ▶ Revisit the `RTC_EPOCH_READ` and `RTC_EPOCH_SET` to change the RTC offset. This will allow changing the RTC offset at runtime.
- ▶ Write documentation on how to write an RTC driver and avoid the common pitfalls.
- ▶ Split up the ds1307 frankendriver.

# Questions? Suggestions? Comments?

Alexandre Belloni  
*[alexandre.belloni@bootlin.com](mailto:alexandre.belloni@bootlin.com)*

Slides under CC-BY-SA 3.0  
<https://bootlin.com/pub/conferences/2019/elce/belloni-rtc>